

Программирование в командном процессоре ОС UNIX.

Ветвления и циклы

Щербак Маргарита Романовна

2022

RUDN

Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Задание:

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:
 - `-iinputfile` — прочитать данные из указанного файла;
 - `-ooutputfile` — вывести данные в указанный файл;
 - `-p` шаблон — указать шаблон для поиска;
 - `-C` — различать большие и малые буквы;
 - `-n` — выдавать номера строк.а затем ищет в указанном файле нужные строки, определяемые ключом `-p`.

Я создала файл prog1.sh, в котором буду писать скрипт, открыла текстовый редактор emacs. Также дала созданному файлу право доступа на выполнение (+x).

Считывать данные я буду из файла conf.txt, поэтому просмотрели его содержимое перед выполнением задания.

Создала файл outputf.txt, в который считывала содержимое файла conf.txt.

При выполнении задания взяла в качестве шаблона из файла conf.txt строку №23: man_db.conf, "db" будет служить шаблоном для параметра -C.

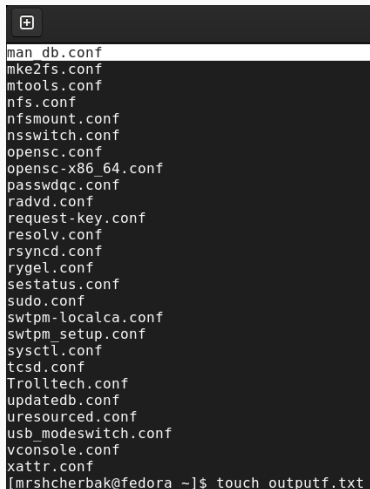
(Рис. 1 - Рис. 5).

Создание файла, в котором написан скрипт первого задания, предоставление права доступа на выполнение этому файлу и просмотр содержимого файла, который будем читать.

```
[mrshcherbak@fedora ~]$ touch prog1.sh
[mrshcherbak@fedora ~]$ emacs &
[1] 36970
[mrshcherbak@fedora ~]$ chmod +x prog1.sh
[mrshcherbak@fedora ~]$ cat conf.txt
anthy-unicode.conf
appstream.conf
asound.conf
brltty.conf
chrony.conf
dleyna-renderer-service.conf
dleyna-server-service.conf
dnsmasq.conf
dracut.conf
extlinux.conf
fprintd.conf
fuse.conf
host.conf
idmapd.conf
jwhois.conf
kdump.conf
krb5.conf
ld.so.conf
libaudit.conf
libuser.conf
locale.conf
logrotate.conf
man_db.conf
mke2fs.conf
mtools.conf
```

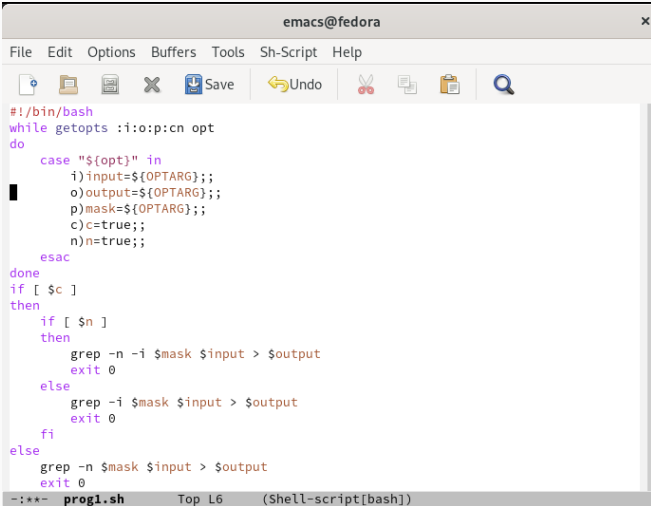
Figure 1: Выполнение

Создание файла, в который будем выводить данные из указанного файла.



```
man db.conf
mke2fs.conf
mttools.conf
nfs.conf
nfsmount.conf
nsswitch.conf
opensc.conf
opencsc-x86_64.conf
passwdqc.conf
radvd.conf
request-key.conf
resolv.conf
rsyncd.conf
rygel.conf
sestatus.conf
sudo.conf
swtpm-localca.conf
swtpm_setup.conf
sysctl.conf
tcsd.conf
Trolltech.conf
updatedb.conf
uresourced.conf
usb_modeswitch.conf
vconsole.conf
xattr.conf
[mrshcherbak@fedora ~]$ touch outputf.txt
```

Figure 2: Выполнение



The image shows a screenshot of an Emacs editor window titled "emacs@fedora". The window has a menu bar with "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". Below the menu bar is a toolbar with icons for opening files, saving, undo, redo, and search. The main text area contains a shell script in bash. The script starts with a shebang line "#!/bin/bash" and a while loop "while getopts :i:o:p:cn opt" that processes command-line options. It uses a case statement to handle options: 'i' sets input, 'o' sets output, 'p' sets mask, 'c' sets c=true, and 'n' sets n=true. After the loop, it checks if 'c' is set, and if so, it checks if 'n' is set. If both are set, it runs "grep -n -i \$mask \$input > \$output" and "exit 0". If only 'c' is set, it runs "grep -i \$mask \$input > \$output" and "exit 0". If neither is set, it runs "grep -n \$mask \$input > \$output" and "exit 0". The status bar at the bottom shows "-:***- prog1.sh Top L6 (Shell-script[bash])".

```
#!/bin/bash
while getopts :i:o:p:cn opt
do
    case "${opt}" in
        i)input=${OPTARG};;
        o)output=${OPTARG};;
        p)mask=${OPTARG};;
        c)c=true;;
        n)n=true;;
    esac
done
if [ $c ]
then
    if [ $n ]
    then
        grep -n -i $mask $input > $output
        exit 0
    else
        grep -i $mask $input > $output
        exit 0
    fi
else
    grep -n $mask $input > $output
    exit 0
fi
```

Figure 3: Скрипт 1го задания

Делаем вывод, что скрипт работает корректно. Командный файл анализирует командную строку с ключами и ищет в указанном файле нужные строки, определяемые ключом -p.

```
[mrshcherbak@fedora ~]$ grep -n -i "DB" conf.txt
23:man_db.conf
44:update_db.conf
[mrshcherbak@fedora ~]$ grep -n "DB" conf.txt
```

Figure 4: Выполнение

```
[mrshcherbak@fedora ~]$ ./prog1.sh -i conf.txt -o outputf.txt -p "DB" -n -c
[mrshcherbak@fedora ~]$ cat outputf.txt
23:man_db.conf
44:update_db.conf
[mrshcherbak@fedora ~]$ ./prog1.sh -i conf.txt -o outputf.txt -p "DB" -c

[mrshcherbak@fedora ~]$ cat outputf.txt
man_db.conf
update_db.conf
[mrshcherbak@fedora ~]$ ./prog1.sh -i conf.txt -o outputf.txt -p "DB"
[mrshcherbak@fedora ~]$ cat outputf.txt
[mrshcherbak@fedora ~]$ ./prog1.sh -i conf.txt -o outputf.txt -p "DB" -n -c
[mrshcherbak@fedora ~]$ cat outputf.txt
23:man_db.conf
44:update_db.conf
```

Figure 5: Проверка работы скрипта с параметрами (ключами)

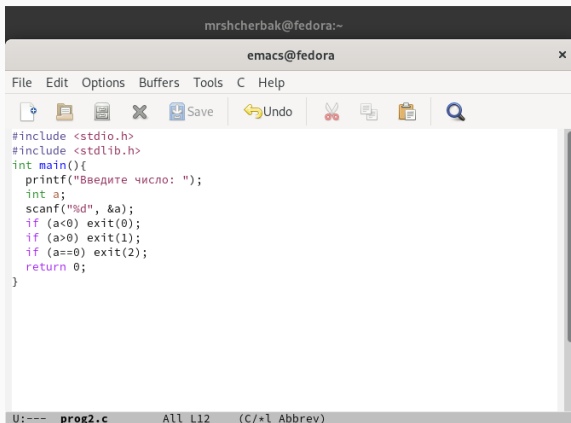
2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию о коде завершения в оболочку. Командный файл должен вызывать эту программу и выдавать сообщение о том, какое число было введено.

Создала файл prog2.sh, в котором писала второй скрипт, и открыла его в редакторе emacs. Также создала файл prog2.c, в котором писала на языке Си программу, выводящую число и определяющую, является ли оно больше нуля, меньше нуля или равно нулю. (Рис. 6 - Рис. 9).

Предоставила право доступа на выполнение файлу prog2.sh.

```
[mrshcherbak@fedora ~]$ touch prog2.sh  
[1]+  Завершён      emacs  
[mrshcherbak@fedora ~]$ emacs &  
[1] 39398  
[mrshcherbak@fedora ~]$ touch prog2.c  
[mrshcherbak@fedora ~]$ chmod +x prog2.sh
```

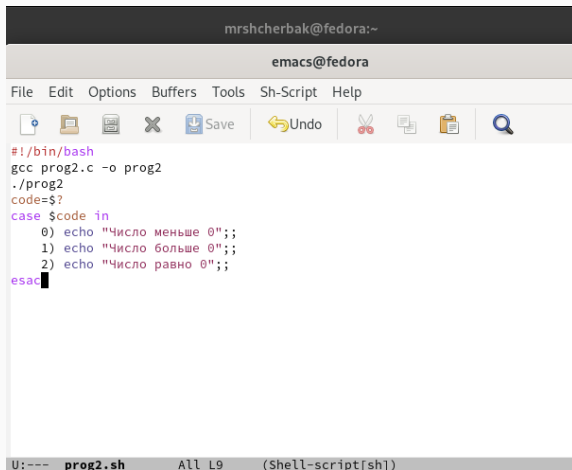
Figure 6: Создание файла, в котором написан скрипт второго задания, предоставление права доступа на выполнение этому файлу



The image shows a terminal window with the username 'mrshcherbak@fedora:~' and an Emacs editor window titled 'emacs@fedora'. The Emacs window has a menu bar with 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'C', and 'Help'. Below the menu bar is a toolbar with icons for creating a new file, opening a file, saving a file, undoing, redoing, and searching. The main text area contains a C program. The status bar at the bottom shows 'U:--- prog2.c All L12 (C/*l Abbrev)'.

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    printf("Введите число: ");
    int a;
    scanf("%d", &a);
    if (a<0) exit(0);
    if (a>0) exit(1);
    if (a==0) exit(2);
    return 0;
}
```

Figure 7: Программа на языке Си



The screenshot shows an Emacs editor window titled 'emacs@fedora'. The window has a menu bar with 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. Below the menu bar is a toolbar with icons for file operations (new, open, save, close, save all), editing (undo, redo, cut, copy, paste), and search. The main text area contains a shell script with the following content:

```
#!/bin/bash
gcc prog2.c -o prog2
./prog2
code=?
case $code in
  0) echo "Число меньше 0";;
  1) echo "Число больше 0";;
  2) echo "Число равно 0";;
esac
```

The status bar at the bottom of the window displays 'U:--- prog2.sh', 'All L9', and '(Shell-script[sh])'.

Figure 8: Скрипт 2го задания

```
[mrshcherbak@fedora ~]$ ./prog2.sh
Введите число: 16
Число больше 0
[mrshcherbak@fedora ~]$ ./prog2.sh
Введите число: 0
Число равно 0
[mrshcherbak@fedora ~]$ ./prog2.sh
Введите число: -69
Число меньше 0
[mrshcherbak@fedora ~]$
```

Figure 9: Выполнение

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).

Создала файл prog3.sh, в котором писала третий скрипт, и открыла его в редакторе emacs (С-х С-с). Предоставила право доступа на выполнение файлу prog3.sh. (Рис. 11).

Написала командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Этот же командный файл удаляет все созданные им файлы (Рис. 10).

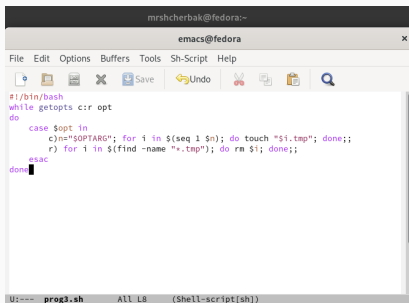
The image shows a screenshot of the Emacs text editor window. The window title is 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The toolbar contains icons for file operations and editing. The main text area displays a shell script named 'prog3.sh'. The script starts with a shebang line '#!/bin/bash' and a 'while' loop that reads command-line options. Inside the loop, there is a 'case' statement for the '-c' option that uses 'seq' to create a series of temporary files (1.tmp to N.tmp) and then removes them. The script ends with a 'done' statement for the 'while' loop. The status bar at the bottom shows 'U:--- prog3.sh All L8 (Shell-script[sh])'.

Figure 10: Скрипт 3го задания

Скрипт работает корректно.

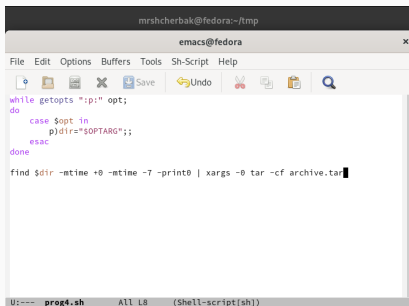
```
[mrshcherbak@fedora ~]$ touch prog3.sh
[mrshcherbak@fedora ~]$ chmod +x prog3.sh
[mrshcherbak@fedora ~]$ ./prog3.sh -c 7
[mrshcherbak@fedora ~]$ ls
1.tmp  acd.txt      exp.doc      file11.txt- Lab007      play      prog3.sh      skript3.sh- Видео
2.tmp  adc.cpp      exp.jpg      file12.txt- lab07.sh   prog1.sh- prog3.sh- skript4.sh- Документы
3.tmp  adc.txt      exp.pdf      file12.txt- lab07.sh- prog1.sh- project.txt skript4.sh- Загрузки
4.tmp  australia   exp.png      file13.txt- may        prog2      reports      text.cpp   Изображения
5.tmp  backup      exp.svg      file13.txt- monthly   prog2.c     ski.places  text.txt   Музыка
6.tmp  backup.sh-  feathers     file14.txt- my_os      prog2.c-    skript2.sh vvvvvvvvvv Общедоступные
7.tmp  bin         '#file11.txt#' file14.txt- otchet.txt prog2.sh- skript2.sh- week3.txt  'Рабочий стол'
abcl   conf.txt    file11.txt  file.txt   outputf.txt prog2.sh- skript3.sh  work       Шаблоны
[mrshcherbak@fedora ~]$ ./prog3.sh -r
[mrshcherbak@fedora ~]$ ls
abcl   conf.txt    file11.txt  file.txt   outputf.txt prog2.sh- skript3.sh  work       Шаблоны
acd.txt  exp.doc      file11.txt- Lab007      play      prog3.sh   skript3.sh- Видео
adc.cpp  exp.jpg      file12.txt- lab07.sh   prog1.sh- prog3.sh- skript4.sh- Документы
adc.txt  exp.pdf      file12.txt- lab07.sh- prog1.sh- project.txt skript4.sh- Загрузки
australia exp.png      file13.txt- may        prog2      reports      text.cpp   Изображения
backup   exp.svg      file13.txt- monthly   prog2.c     ski.places  text.txt   Музыка
backup.sh- feathers     file14.txt- my_os      prog2.c-    skript2.sh vvvvvvvvvv Общедоступные
bin      '#file11.txt#' file14.txt- otchet.txt prog2.sh- skript2.sh- week3.txt  'Рабочий стол'
[mrshcherbak@fedora ~]$
```

Figure 11: Проверка выполнения скрипта

4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`).

Создала файл prog4.sh, в котором писала четвёртый скрипт, и открыла его в редакторе emacs (C-x C-s). Предоставила право доступа на выполнение файлу prog4.sh. (Рис. 13).

Написала командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Запаковываются те файлы, которые были изменены менее недели тому назад. (Рис. 12).



```
mrshcherbak@fedora:~/tmp
emacs@fedora
File Edit Options Buffers Tools Sh-Script Help
[Icons] Save Undo [Icons]
while getopts ":p:" opt;
do
  case $opt in
    p)dir="$OPTARG";;
    esac
  done
find $dir -mtime +0 -mtime -7 -print0 | xargs -0 tar -cf archive.tar
U:--- prog4.sh All L8 (Shell-script[sh])
```

Figure 12: Скрипт 4го задания

В папке tmp, созданной в домашнем каталоге, находятся файлы, изменённые менее недели тому назад.

```
[mrshcherbak@fedora ~]$ touch prog4.sh
[mrshcherbak@fedora ~]$ chmod +x prog4.sh
[mrshcherbak@fedora ~]$ ./prog4.sh /home/mrshcherbak
[mrshcherbak@fedora ~]$ ls
abcl      bin      '#file11.txt#'  file14.txt~  otchet.txt  prog2.sh
acd.txt   conf.txt  file11.txt      file.txt     outputf.txt  prog2.sh~
adc.cpp   exp.doc   file11.txt~     Lab007       play        prog3.sh
adc.txt   exp.jpg   file12.txt      lab07.sh     prog1.sh~   prog3.sh~
archive.tar exp.pdf   file12.txt~     lab07.sh~    prog1.sh~   prog4.sh
australia exp.png   file13.txt      may          prog2       prog4.sh~
backup    exp.svg   file13.txt~     monthly     prog2.c     project.txt
backup.sh~ feathers  file14.txt      my_os       prog2.c~    reports

[mrshcherbak@fedora ~]$ mkdir tmp
[mrshcherbak@fedora ~]$ tar -xf archive.tar -C /home/mrshcherbak/tmp
[mrshcherbak@fedora ~]$ cd tmp
[mrshcherbak@fedora tmp]$ ls
backup    exp.doc   exp.pdf  exp.svg  skript2.sh  skript3.sh  skript4.sh
backup.sh~ exp.jpg   exp.png  otchet.txt skript2.sh~ skript3.sh~ skript4.sh~
[mrshcherbak@fedora tmp]$
```

Figure 13: Проверка работы скрипта

Таким образом, в ходе ЛРН^{№11} я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.